

Министерство образования и науки РФ
ФГБОУ ВО ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Кафедра «Информационная безопасность систем и технологий»

ОТЧЕТ
о практической работе №2
ИСПОЛЬЗОВАНИЕ СТАНДАРТНОЙ БИБЛИОТЕКИ СИ++

Дисциплина: Языки программирования

Группа: 18ПИ1

Выполнил: Новиков Д.О.

Количество баллов:

Дата сдачи:

Принял: к.т.н., доцент Лупанов М. Ю.

Пенза 2019

1 Цель работы

1.1 Знакомство с возможностями и использование стандартной библиотеки Си++.

2 Задание к лабораторной работе

2.1 Напишите программу, получающую значение текущего времени и выводящего его в том же формате, что и функция `ctime`, но русскими названиями дней недели и месяцев.

2.2 Напишите программу, формирующую три целочисленных вектора размером 10000000 элементов и заполните их случайными значениями в диапазоне от -1000000000 до 1000000000. Первый вектор должен создаваться пустым и элементы должны добавляться в цикле в конец вектора. Второй вектор должен создаваться сразу нужного размера и заполняться с помощью алгоритма `generate`. Третий вектор должен быть создан как копия второго.

2.3 Используя возможности библиотеки `chrono` добавьте в предыдущую программу код для определения времени создания и заполнения каждого из трех векторов. Соберите программу в конфигурации «Release» и выполните несколько раз. Поясните полученные результаты замеров времени.

2.4 Добавьте в предыдущую программу сортировку второго и третьего векторов. Один вектор отсортируйте с помощью алгоритма `sort`, второй с помощью алгоритма `stable_sort`. Добавьте код для определения времени сортировки и проведите эксперимент. Поясните полученные результаты замеров времени.

2.5 Разработайте структуру данных для моделирования колоды карт (36 или 52, на ваш выбор). Напишите код, выполняющий следующие действия: заполнение колоды, перемешивание колоды, поиск в колоде двух подряд карт одного цвета, поиск в колоде двух подряд карт одного номинала, поиск в колоде дамы пик, поиск в колоде всех тузов, печать колоды.

2.6 Разработайте программу, читающую из файла список людей в формате «Фамилия Имя Отчество» и выполняющий с ним следующие действия: сортировка по фамилии, поиск однофамильцев, поиск самого редкого имени(имен), поиск самого популярного имени (имен).

3 Результаты работы

3.1 Чтобы функция `ctime` выводила значение текущего времени и названия дней недели и месяцев на русском языке, нужно использовать оператор `switch`, который обрабатывает и выводит полученные данные с `tm_wday`, дней недели от 0 до 6, и `tm_mon`, месяцев от 0 до 11. Пример работы программы представлен на рисунке 1. Полный текст представлен в приложении А.

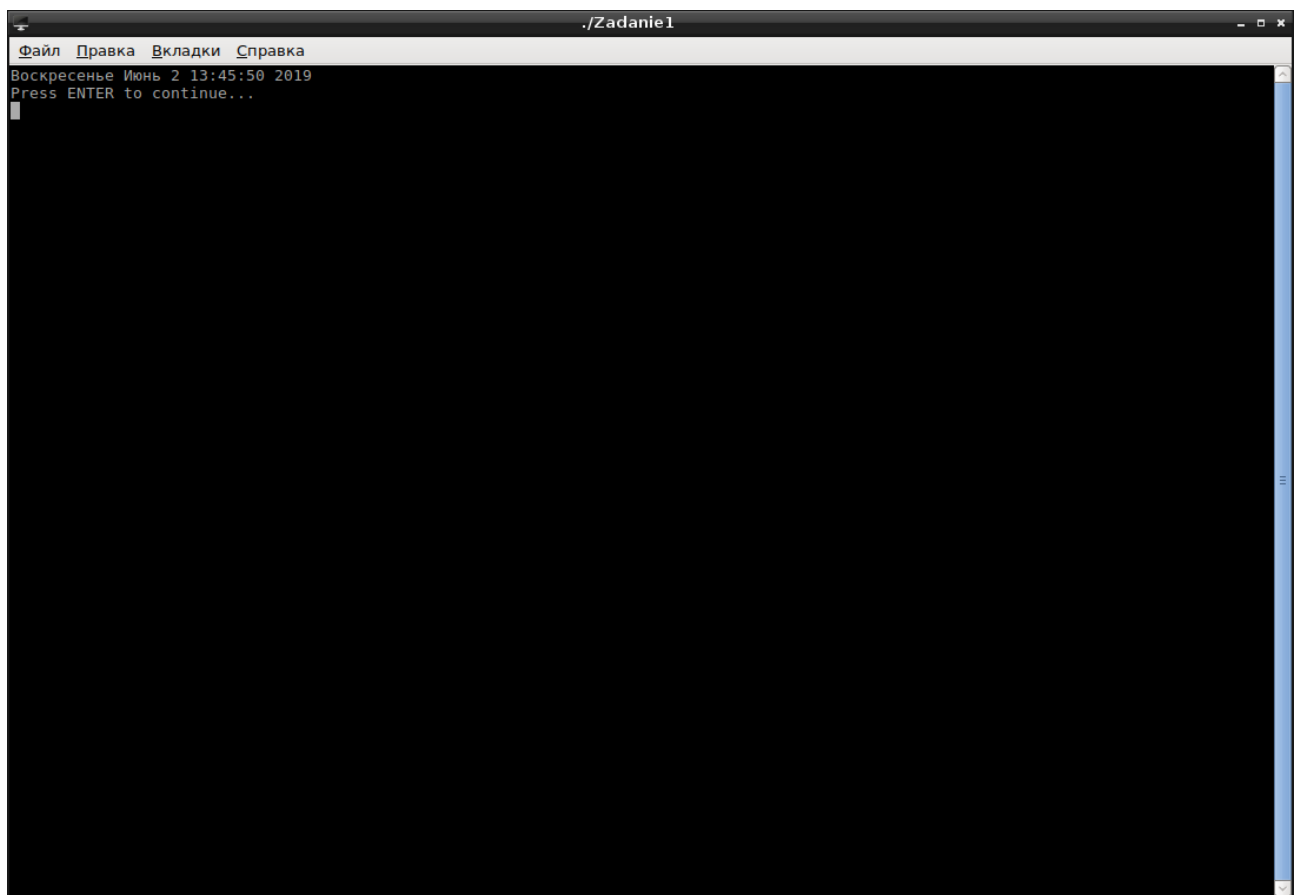
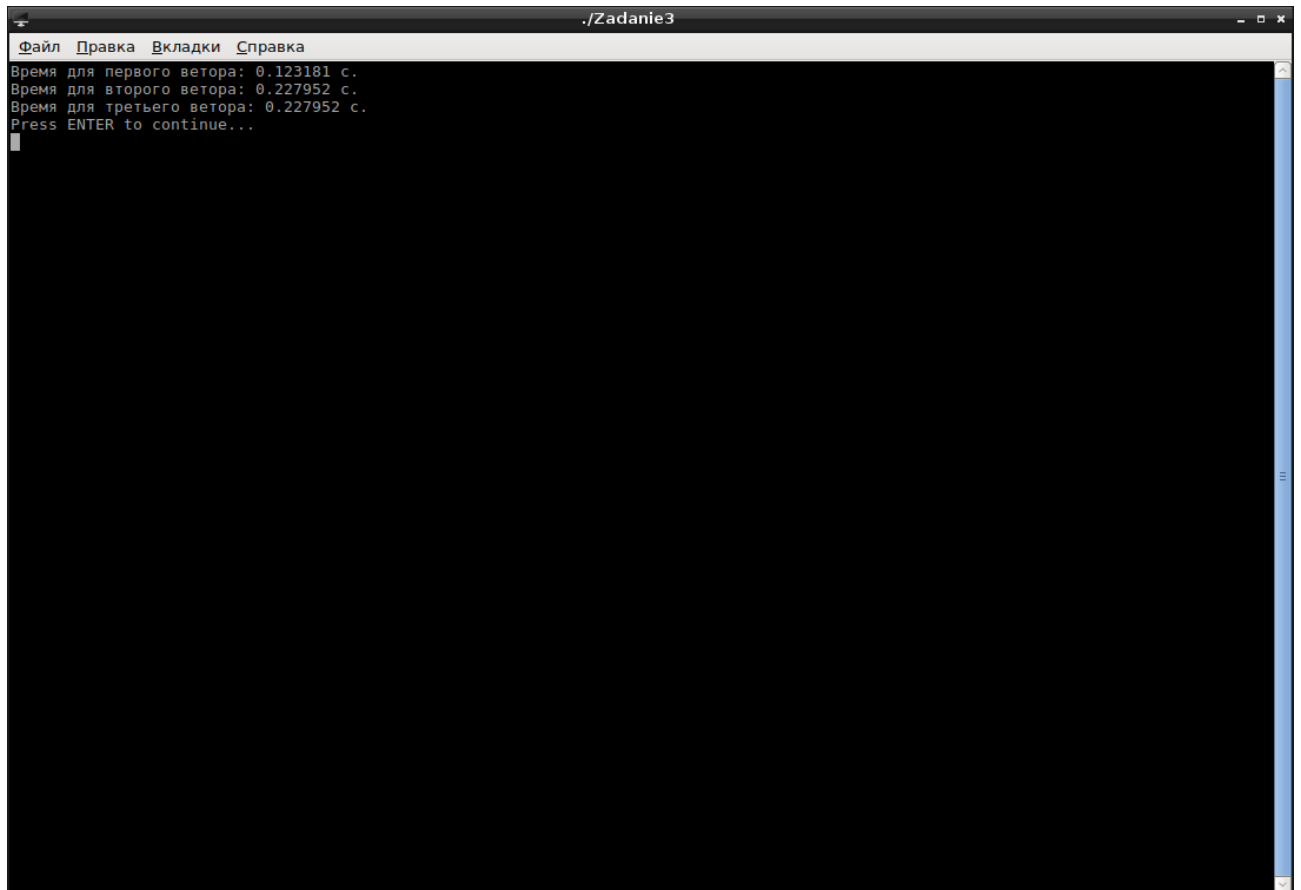


Рисунок 1 - Результат работы программы 1

3.2 Первый вектор создали пустым и элементы добавляются в цикле в конец вектора с помощью метода `push_back()`. Второй вектор создали сразу нужного размера, то есть на 10000000 элементов, и заполняли его с помощью

алгоритма generate. Третий вектор создали как копию второго. Пример работы программы представлен на рисунке 2. Полный текст представлен в приложении Б.

Рисунок 2 - Результат работы программы 2



```
./Zadanie3
Файл  Правка  Вкладки  Справка
Время для первого вектора: 0.123181 с.
Время для второго вектора: 0.227952 с.
Время для третьего вектора: 0.227952 с.
Press ENTER to continue...
```

Рисунок 3 - Результат работы программы 3

При выполнении программы несколько раз, мы заметили, что 2 и 3 вектор заполняются всегда с одинаковым временем.

3.4 Для программы, описанной в пункте 3.3, мы добавили сортировку второго и третьего векторов, а также определили время каждой сортировки. Один вектор отсортировали с помощью алгоритма `sort`, второй с помощью алгоритма `stable_sort`. Пример работы программы представлен на рисунке 4. Полный текст представлен в приложении Г.

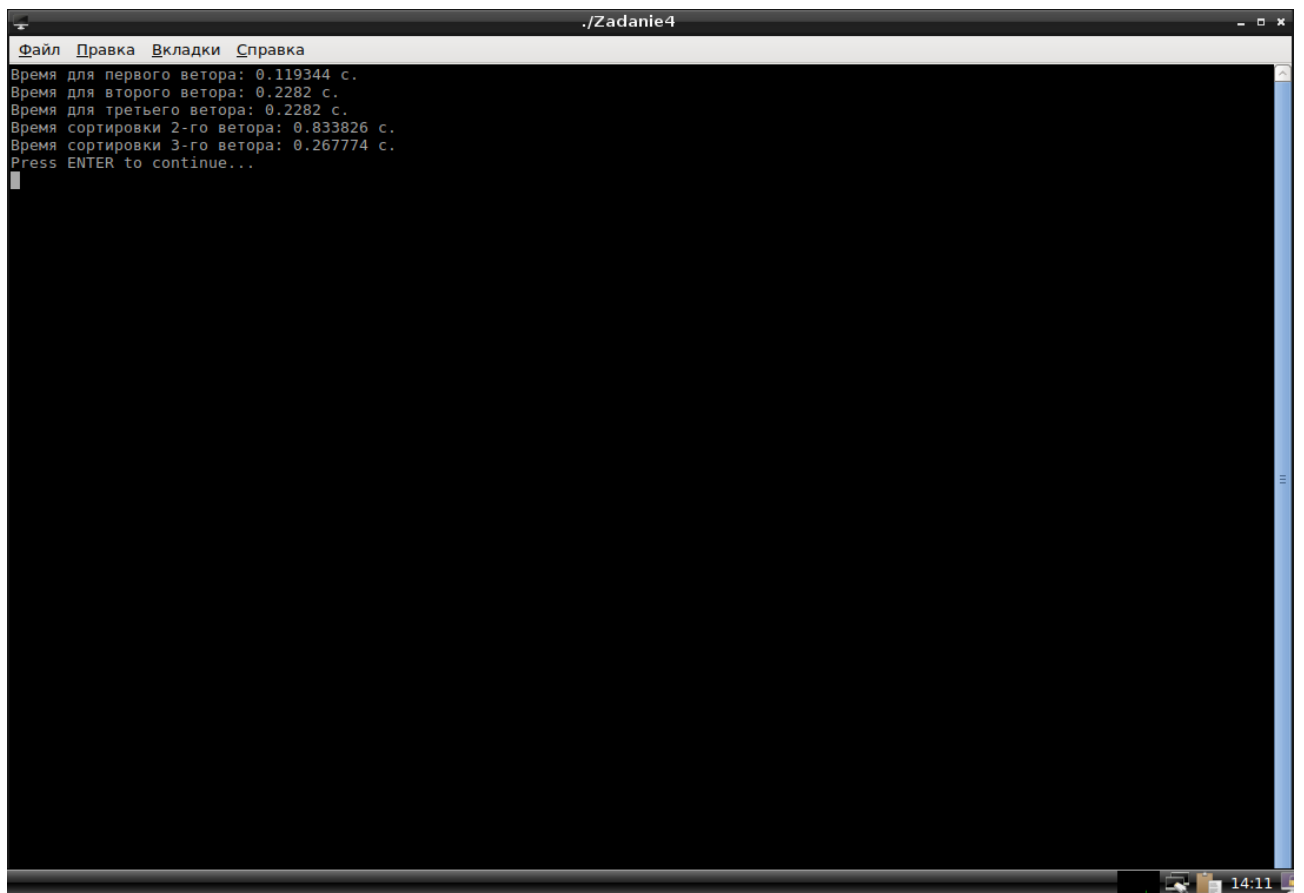
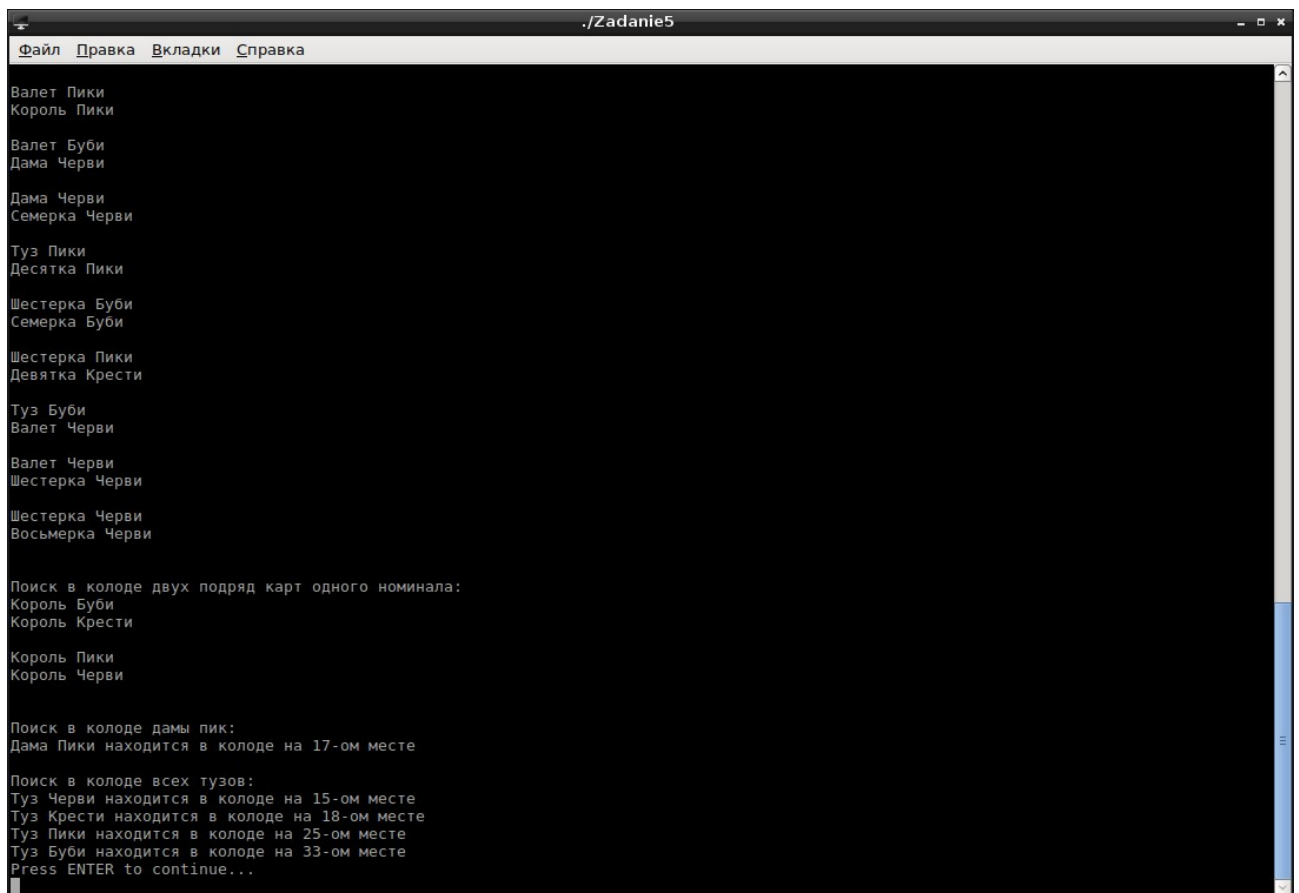


Рисунок 4 - Результат работы программы 4

Из результатов выполнения программы видно, что сортировка `stable_sort` затрачивает значительно меньше времени, чем обычная сортировка `sort`.

3.5 Для программы была выбрана колода из 36 карт. Была создана структура с двумя переменными типа `integer` `real_mast` и `real_number`, значение масти и номинала соответственно. Для перемешивания колоды использовался алгоритм `shuffle` с использованием генератора ПСП `mt19937`. Пример работы программы представлен на рисунке 5. Полный текст представлен в приложении Д.



```
./Zadanie5
Файл  Правка  Вкладки  Справка

Валет Пики
Король Пики

Валет Буби
Дама Черви

Дама Черви
Семерка Черви

Туз Пики
Десятка Пики

Шестерка Буби
Семерка Буби

Шестерка Пики
Девятка Крести

Туз Буби
Валет Черви

Валет Черви
Шестерка Черви

Шестерка Черви
Восьмерка Черви

Поиск в колоде двух подряд карт одного номинала:
Король Буби
Король Крести

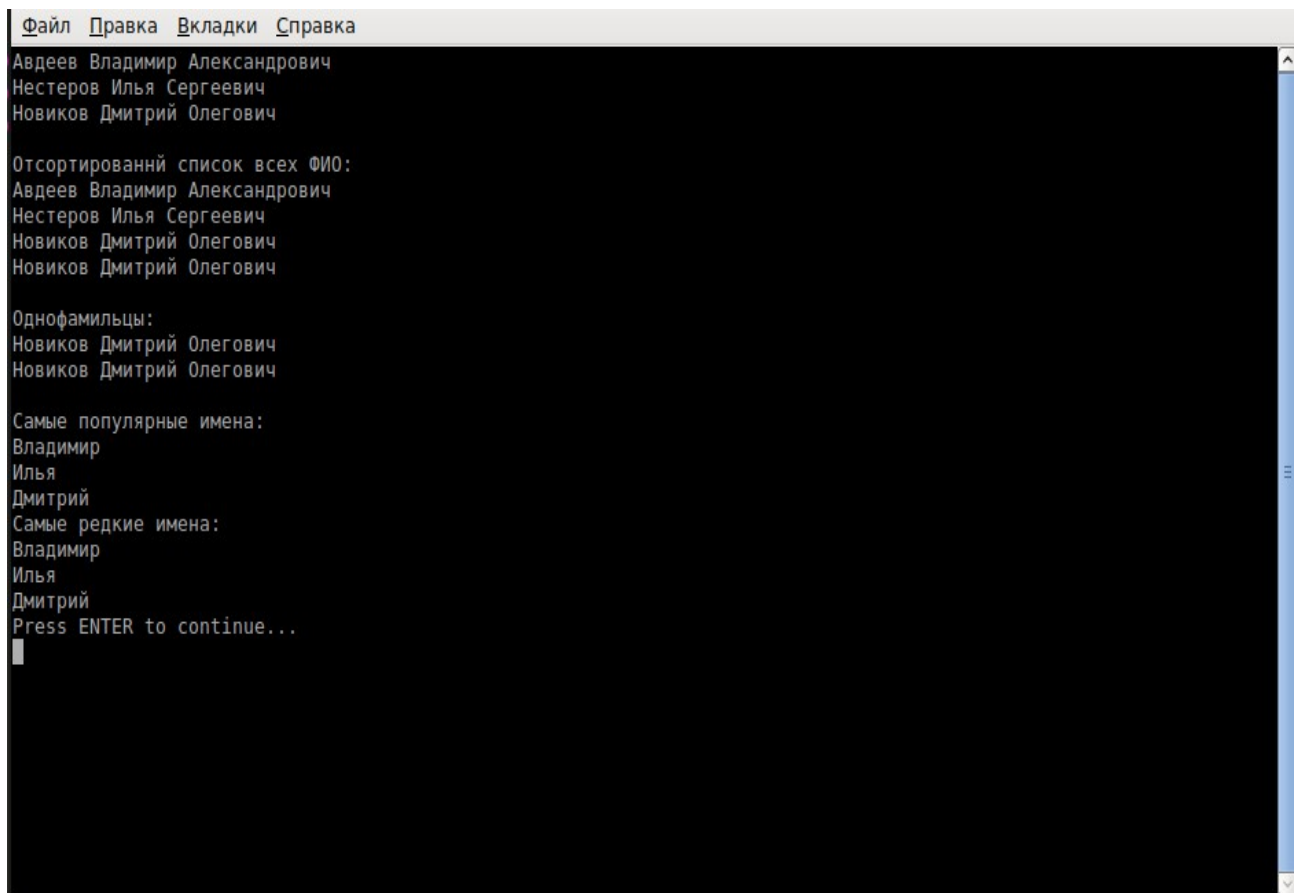
Король Пики
Король Черви

Поиск в колоде дамы пик:
Дама Пики находится в колоде на 17-ом месте

Поиск в колоде всех тузов:
Туз Черви находится в колоде на 15-ом месте
Туз Крести находится в колоде на 18-ом месте
Туз Пики находится в колоде на 25-ом месте
Туз Буби находится в колоде на 33-ом месте
Press ENTER to continue...
```

Рисунок 5 - Результат работы программы 5

3.6 В программе ФИО считываются из файла и заносятся в вектор с помощью метода `push_back()`. Для сортировки вектора по фамилиям использовалась сортировка пузырьком. Пример работы программы представлен на рисунке 6. Полный текст представлен в приложении Е.

A screenshot of a text editor window with a menu bar containing 'Файл', 'Правка', 'Вкладки', and 'Справка'. The editor displays the output of a C++ program. The text is as follows:

```
Авдеев Владимир Александрович
Нестеров Илья Сергеевич
Новиков Дмитрий Олегович

Отсортированный список всех ФИО:
Авдеев Владимир Александрович
Нестеров Илья Сергеевич
Новиков Дмитрий Олегович
Новиков Дмитрий Олегович

Однофамильцы:
Новиков Дмитрий Олегович
Новиков Дмитрий Олегович

Самые популярные имена:
Владимир
Илья
Дмитрий
Самые редкие имена:
Владимир
Илья
Дмитрий
Press ENTER to continue...
█
```

Рисунок 6 - Результат работы программы 6

4 Вывод

В результате выполнения работы было изучено использование стандартной библиотеки Си++, а также была написана программа для заполнения колоды карт, и получены практические навыки в использовании стандартных алгоритмов языка с++.

Приложение А.

Текст программы функции ctime на русском

```
#include <iostream>
#include <ctime>
using namespace std;

void get_time(tm * ptm);

int main(int argc, char **argv)
{
    time_t t = time(NULL);
    tm * ptm;
    ptm = localtime(&t);
    get_time(ptm);
    return 0;
}

void get_time(tm * ptm)
{
    string s;
    switch (ptm->tm_wday) {
    case 0:
        s += "Воскресенье";
        break;
    case 1:
        s += "Понедельник";
        break;
    case 2:
        s += "Вторник";
        break;
    case 3:
        s += "Среда";
```

```

        break;
case 4:
    s += "Четверг";
    break;
case 5:
    s += "Пятница";
    break;
case 6:
    s += "Суббота";
    break;
}

```

```

switch (ptm->tm_mon) {
case 0:
    s += " Январь";
    break;
case 1:
    s += " Февраль";
    break;
case 2:
    s += " Март";
    break;
case 3:
    s += " Апрель";
    break;
case 4:
    s += " Май";
    break;
case 5:
    s += " Июнь";
    break;
case 6:
    s += " Июль";

```

```

        break;
    case 7:
        s += " Август";
        break;
    case 8:
        s += " Сентябрь";
        break;
    case 9:
        s += " Октябрь";
        break;
    case 10:
        s += " Ноябрь";
        break;
    case 11:
        s += " Декабрь";
        break;
    }

    cout << s << " " << ptm->tm_mday << " " << ptm->tm_hour << ":"
<< ptm->tm_min << ":" << ptm->tm_sec << " " << ptm->tm_year+1900
<< endl;
}

```

Приложение Б.

Текст программы заполнения вектора случайными значениями

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <random>
using namespace std;

int RandomGenerator()
{
    static mt19937 rnd((uint64_t)&rnd);
    uniform_int_distribution<int> d(-1000000000, 1000000000);
    return d(rnd);
}

int main()
{
    vector<int> v;
    mt19937 rnd(-1000000000);
    for (int i=0; i < 10000000; i++)
        v.push_back(rnd());
    vector<int> v1(10000000);
    generate(v1.begin(), v1.end(), RandomGenerator);
    vector<int> v2(v1);
    for(auto e:v2)
        cout<<e<<'\t';
    cout<<endl;
}
```

Приложение В.

Текст программы для подсчета времени заполнения векторов

```
#include <algorithm>
#include <vector>
#include <random>
#include <chrono>
using namespace std;
using namespace std::chrono;

int RandomGenerator()
{
    static mt19937 rnd((uint64_t)&rnd);
    uniform_int_distribution<int> d(-1000000000, 1000000000);
    return d(rnd);
}

int main()
{
    random_device rd;
    vector<int> v;
    mt19937 rnd(-1000000000);
    steady_clock::time_point tp1 = steady_clock::now();
    for (int i=0; i < 10000000; i++)
        v.push_back(rnd());
    steady_clock::time_point tp2 = steady_clock::now();
    duration<double> d = tp2 - tp1;
    cout << "Время для первого ветора: " << d.count() << " с." <<
endl;
    vector<int> v1(10000000);
    tp1 = steady_clock::now();
    generate(v1.begin(), v1.end(), RandomGenerator);
    tp2 = steady_clock::now();
    d = tp2 - tp1;
```

```
        cout << "Время для второго ветора: " << d.count() << " с." <<
endl;
        tp1 = steady_clock::now();
        vector<int> v2(v1);
        tp2 = steady_clock::now();
        cout << "Время для третьего ветора: " << d.count() << " с." <<
endl;
    }
```

Приложение Г.

Текст программы для сортировки векторов

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <random>
#include <chrono>
using namespace std;
using namespace std::chrono;

int RandomGenerator()
{
    static mt19937 rnd((uint64_t)&rnd);
    uniform_int_distribution<int> d(-1000000000, 1000000000);
    return d(rnd);
}

int main()
{
    random_device rd;
    vector<int> v;
    mt19937 rnd(-1000000000);
    steady_clock::time_point tp1 = steady_clock::now();
    for (int i=0; i < 10000000; i++)
        v.push_back(rnd());
    steady_clock::time_point tp2 = steady_clock::now();
    duration<double> d = tp2 - tp1;
    cout << "Время для первого ветопа: " << d.count() << " с." <<
endl;
    vector<int> v1(10000000);
    tp1 = steady_clock::now();
    generate(v1.begin(), v1.end(), RandomGenerator);
    tp2 = steady_clock::now();
```

```

    d = tp2 - tp1;
    cout << "Время для второго ветора: " << d.count() << " с." <<
endl;
    tp1 = steady_clock::now();
    vector<int> v2(v1);
    tp2 = steady_clock::now();
    cout << "Время для третьего ветора: " << d.count() << " с." <<
endl;
    tp1 = steady_clock::now();
    sort(v1.begin(),v1.end());
    tp2 = steady_clock::now();
    d = tp2 - tp1;
    cout << "Время сортировки 2-го ветора: " << d.count() << " с."
<< endl;
    tp1 = steady_clock::now();
    stable_sort(v1.begin(), v1.end());
    tp2 = steady_clock::now();
    d = tp2 - tp1;
    cout << "Время сортировки 3-го ветора: " << d.count() << " с."
<< endl;
}

```


Приложение Д.

Текст программы для колоды карт

```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>
#include <random>

using namespace std;

string    number[9]    {"Шестерка",    "Семерка",    "Восьмерка",
"Девятка","Десятка", "Валет", "Дама", "Король", "Туз"};
string mast[4] {"Крести", "Черви", "Буби", "Пики"};

struct karta {
    int real_mast;
    int real_number;
};

void fill_koloda(vector<karta>& t);
void mix_koloda(vector<karta>& t);
void find_color(vector<karta>& t); //Для первого вхождения в колоде
void find_mast(vector<karta>& t); //Для первого вхождения в колоде
void find_dama(vector<karta>& t);
void print_ace(vector<karta>& t);
void print_koloda(vector<karta>& t);

int main()
{
    vector<karta> koloda36(36);
    cout << "Заполнение колоды:" << endl;
    fill_koloda(koloda36);
    print_koloda(koloda36);
}
```

```

    cout << endl;
    cout << "Перемешивание колоды:" << endl;
    mix_koloda(koloda36);
    print_koloda(koloda36);
    cout << endl;
    cout << "Поиск в колоде двух подряд карт одного цвета:" <<
endl;
    find_color(koloda36);
    cout << endl;
    cout << "Поиск в колоде двух подряд карт одного номинала:" <<
endl;
    find_mast(koloda36);
    cout << endl;
    cout << "Поиск в колоде дамы пик:" << endl;
    find_dama(koloda36);
    cout << endl;
    cout << "Поиск в колоде всех тузов:" << endl;
    print_ace(koloda36);
    return 0;
}

void fill_koloda(vector<karta>& t)
{
    int i;
    int mast1 = 0;
    int number1 = 6;
    for(i = 0; i < 36; i++) {
        t[i].real_number = number1;
        t[i].real_mast = mast1;
        if (mast1 == 3) {
            mast1 = 0;
            number1++;
        } else

```

```

        mast1++;
    }
}

void mix_koloda(vector<karta>& t)
{
    static mt19937 rnd((uint64_t)&rnd);
    shuffle(t.begin(), t.end(), rnd);
}

void find_color(vector<karta>& t)
{
    int i;
    for(i = 1; i < 36; i++) {
        if ((t[i].real_mast == 1 && t[i-1].real_mast == 2) ||
            (t[i].real_mast == 0 && t[i-1].real_mast == 3) ||
            (t[i].real_mast == 1 && t[i-1].real_mast == 1) ||
            (t[i].real_mast == 2 && t[i-1].real_mast == 2) ||
            (t[i].real_mast == 3 && t[i-1].real_mast == 3) ||
            (t[i].real_mast == 4 && t[i-1].real_mast == 4))
        {
            cout << number[t[i - 1].real_number - 6] << " " <<
            mast[t[i - 1].real_mast] << endl;
            cout << number[t[i].real_number - 6] << " " <<
            mast[t[i].real_mast] << endl;
            cout << endl;
        }
    }
}

void find_mast(vector<karta>& t)
{
    int i;

```

```

        for(i = 1; i < 36; i++) {
            if (t[i].real_number == t[i-1].real_number) {
                cout << number[t[i] - 1].real_number - 6] << " " <<
mast[t[i] - 1].real_mast] << endl;
                cout << number[t[i].real_number - 6] << " " <<
mast[t[i].real_mast] << endl;
                cout << endl;
            }
        }
    }

void find_dama(vector<karta>& t)
{
    int i;
    for(i = 0; i < 36; i++) {
        if (t[i].real_number == 12 && t[i].real_mast == 3) {
            cout << number[t[i].real_number - 6] << " " <<
mast[t[i].real_mast] << " находится в колоде на " << i + 1 << "-ом
месте" << endl;
            break;
        }
    }
}

void print_ace(vector<karta>& t)
{
    int i;
    for(i = 0; i < 36; i++) {
        if (t[i].real_number == 14 && t[i].real_mast == 0) {
            cout << number[t[i].real_number - 6] << " " <<
mast[t[i].real_mast] << " находится в колоде на " << i + 1 << "-ом
месте" << endl;
        }
    }
}

```

```

        if (t[i].real_number == 14 && t[i].real_mast == 1) {
            cout << number[t[i].real_number - 6] << " " <<
mast[t[i].real_mast] << " находится в колоде на " << i + 1 << "-ом
месте" << endl;
        }
        if (t[i].real_number == 14 && t[i].real_mast == 2) {
            cout << number[t[i].real_number - 6] << " " <<
mast[t[i].real_mast] << " находится в колоде на " << i + 1 << "-ом
месте" << endl;
        }
        if (t[i].real_number == 14 && t[i].real_mast == 3) {
            cout << number[t[i].real_number - 6] << " " <<
mast[t[i].real_mast] << " находится в колоде на " << i + 1 << "-ом
месте" << endl;
        }
    }
}

void print_koloda(vector<karta>& t)
{
    int i;
    for(i = 0; i < 36; i++) {
        cout << number[t[i].real_number - 6] << " " <<
mast[t[i].real_mast] << endl;;
    }
}

```

Приложение Е.

Текст программы для считывания из файла списка людей

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <vector>
using namespace std;

void bubblesort(vector<string>& mass, unsigned size);
void find_surname(vector<string>& mass, unsigned size);
void find_name(vector<string>& mass, int size);
int space(string s, unsigned begin = 0);

int main(int argc, char **argv)
{
    unsigned i = 0;
    ifstream f("/root/Laba9/Zadanie6/data_v1");
    vector<string> s;
    string line;
    while (getline(f, line)) {
        if (line != "" || line != " " || line != "\n")
            s.push_back(line);
    }
    cout << "Список всех ФИО:" << endl;
    for (i = 0; i < s.size(); i++)
        cout << s[i] << endl;
    cout << endl;
    bubblesort(s, s.size());
    cout << "Отсортированный список всех ФИО:" << endl;
    for (i = 0; i < s.size(); i++)
        cout << s[i] << endl;
    cout << endl;
```

```

        cout << "Однофамильцы:" << endl;
        find_surname(s, s.size());
        cout << endl;
        find_name(s, s.size());
        f.close();
        return 0;
    }

void bubblesort(vector<string>& mass, unsigned size)
{
    unsigned i,j;
    bool swapped;
    for(i = 0; i < size-1; i++) {
        swapped = false;
        for(j = 0; j < size-i-1; j++) {
            if (mass[j] > mass[j + 1]) {
                string Temp = mass[j];
                mass[j] = mass[j + 1];
                mass[j + 1] = Temp;
                swapped = true;
            }
        }
        if (swapped == false)
            break;
    }
}

void find_surname(vector<string>& mass, unsigned size)
{
    string surname1, surname2;
    int probel;
    unsigned i = 0, j = 0;
    for(i = 0; i < size; i++) {

```

```

        probel = space(mass[i]);
        surname1 = mass[i].substr(0,probel);
        for(j = i + 1; j < size; j++) {
            probel = space(mass[j]);
            surname2 = mass[j].substr(0,probel);
            if (surname1 == surname2) {
                cout << mass[i] << endl;
                cout << mass[j] << endl;
            }
        }
    }
}

int space(string s, unsigned begin)
{
    int i = 0;
    if (begin > 0) {
        begin++;
        i++;
    }
    while (s[begin] != ' ') {
        i++;
        begin++;
    }
    return i;
}

void find_name(vector<string>& mass, int size)
{
    string name1, name2;
    int probel1, probel2;
    int i = 0, j = 0, max = -1, min = size + 1;
    int name_count[size];

```



```

for (i = 0; i < size; i++)
    name_count[i] = 0;
for (i = 0; i < size; i++) {
    probell1 = space(mass[i]);
    probel2 = space(mass[i], probell1);
    name1 = mass[i].substr(probell1, probel2);
    for (j = 0; j < size; j++) {
        probell1 = space(mass[j]);
        probel2 = space(mass[j], probell1);
        name2 = mass[j].substr(probell1, probel2);
        if ((mass[i] != mass[j]) && (name1 == name2))
            name_count[i]++;
    }
}

for (i = 0; i < size; i++) {
    if (name_count[i] > max)
        max = name_count[i];
    else if (name_count[i] < min)
        min = name_count[i];
}

//max
name1 = "";
name2 = "";
cout << "Самые популярные имена:" << endl;
for (i = 0; i < size; i++) {
    if (name_count[i] == max) {
        probell1 = space(mass[i]);
        probel2 = space(mass[i], probell1);
        name1 = mass[i].substr(probell1 + 1, probel2);
        if (name1 == name2)
            continue;
        cout << name1 << endl;
        name2 = name1;
    }
}

```

```

        }
    }
    //min
    name1 = "";
    name2 = "";
    cout << "Самые редкие имена:" << endl;
    for (i = 0; i < size; i++) {
        if (name_count[i] == min) {
            probell1 = space(mass[i]);
            probel2 = space(mass[i], probell1);
            name1 = mass[i].substr(probell1 + 1, probel2);
            if (name1 == name2)
                continue;
            cout << name1 << endl;
            name2 = name1;
        }
    }
}

```